TL

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/942,001	08/29/2001	Otto Lodewijk Steinbusch	NL 000477	9027

24738 7590 03/30/2006

PHILIPS ELECTRONICS NORTH AMERICA CORPORATION
INTELLECTUAL PROPERTY & STANDARDS
1109 MCKAY DRIVE, M/S-41SJ
SAN JOSE, CA 95131

EXAMINER	
STEELMAN, MARY J	
ART UNIT	**PAPER NUMBER**
2191	

DATE MAILED: 03/30/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

Office Action Summary	Application No.	Applicant(s)
	09/942,001	STEINBUSCH ET AL.
	Examiner	Art Unit
	Mary J. Steelman	2191

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 January 2006*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-8* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-8* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *14 February 2005* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☒ None of:

1.☒ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This Office Action is in response to RCE, Amendments, and Remarks received 5 January 2006. Per Applicant's request, claims 1 and 5 have been amended. Claims 1-8 are pending.

Priority

2. Acknowledgment is made of applicant's claim for foreign priority based on an application filed in 08/29/2001. Examiner does not have access to a certified copy of the foreign application as required by 35 U.S.C. 119(b). Although correspondence received 1/10/2002 states that the document is attached, it is not scanned. Please submit a duplicate certified copy of the foreign application.

Specification

3. Page 2, line 25 recites "theprogram", should be –the program--. Separate the two words with a space.

4. Examiner objects to the arrangement of the content of the Specification.

`Content of Specification`

(a) Title of the Invention: See 37 CFR 1.72(a) and MPEP § 606. The title of the invention should be placed at the top of the first page of the specification unless the title is provided in an application data sheet. The title of the invention should be brief but technically accurate and descriptive, preferably from two to seven words may not contain more than 500 characters.

(b) Cross-References to Related Applications: See 37 CFR 1.78 and MPEP § 201.11.

(c) Statement Regarding Federally Sponsored Research and Development: See MPEP § 310.

(d) Incorporation-By-Reference Of Material Submitted On a Compact Disc: The specification is required to include an incorporation-by-reference of electronic documents that are to become part of the permanent United States Patent and

Trademark Office records in the file of a patent application. See 37 CFR 1.52(e) and MPEP § 608.05. Computer program listings (37 CFR 1.96(c)), "Sequence Listings" (37 CFR 1.821(c)), and tables having more than 50 pages of text were permitted as electronic documents on compact discs beginning on September 8, 2000.

Or alternatively, Reference to a "Microfiche Appendix": See MPEP § 608.05(a). "Microfiche Appendices" were accepted by the Office until March 1, 2001.

(e) Background of the Invention: See MPEP § 608.01(c). The specification should set forth the Background of the Invention in two parts:

(1) Field of the Invention: A statement of the field of art to which the invention pertains. This statement may include a paraphrasing of the applicable U.S. patent classification definitions of the subject matter of the claimed invention. This item may also be titled "Technical Field."

(2) Description of the Related Art including information disclosed under 37 CFR 1.97 and 37 CFR 1.98: A description of the related art known to the applicant and including, if applicable, references to specific related art and problems involved in the prior art which are solved by the applicant's invention. This item may also be titled "Background Art."

(f) Brief Summary of the Invention: See MPEP § 608.01(d). A brief summary or general statement of the invention as set forth in 37 CFR 1.73. The summary is separate and distinct from the abstract and is directed toward the invention rather than the disclosure as a whole. The summary may point out the advantages of the invention or how it solves problems previously existent in the prior art (and preferably indicated in the Background of the Invention). In chemical cases it should point out in general terms the utility of the invention. If possible, the nature and gist of the invention or the inventive concept should be set forth. Objects of the invention should be treated briefly and only to the extent that they contribute to an understanding of the invention.

(g) Brief Description of the Several Views of the Drawing(s): See MPEP § 608.01(f). A reference to and brief description of the drawing(s) as set forth in 37 CFR 1.74.

(h) Detailed Description of the Invention: See MPEP § 608.01(g). A description of the preferred embodiment(s) of the invention as required in 37 CFR 1.71. The description should be as short and specific as is necessary to describe the invention adequately and accurately. Where elements or groups of elements, compounds, and processes, which are conventional and generally widely known in the field of the invention described and their exact nature or type is not necessary for an understanding and use of the invention by a person skilled in the art, they should not be described in detail. However, where particularly

complicated subject matter is involved or where the elements, compounds, or processes may not be commonly or widely known in the field, the specification should refer to another patent or readily available publication which adequately describes the subject matter.

(i) Claim or Claims: See 37 CFR 1.75 and MPEP § 608.01(m). The claim or claims must commence on separate sheet or electronic page (37 CFR 1.52(b)(3)). Where a claim sets forth a plurality of elements or steps, each element or step of the claim should be separated by a line indentation. There may be plural indentations to further segregate subcombinations or related steps. See 37 CFR 1.75 and MPEP § 608.01(i)-(p).

(j) Abstract of the Disclosure: See MPEP § 608.01(f). A brief narrative of the disclosure as a whole in a single paragraph of 150 words or less commencing on a separate sheet following the claims. In an international application which has entered the national stage (37 CFR 1.491(b)), the applicant need not submit an abstract commencing on a separate sheet if an abstract was published with the international application under PCT Article 21. The abstract that appears on the cover page of the pamphlet published by the International Bureau (IB) of the World Intellectual Property Organization (WIPO) is the abstract that will be used by the USPTO. See MPEP § 1893.03(e).

(k) Sequence Listing, See 37 CFR 1.821-1.825 and MPEP §§ 2421-2431. The requirement for a sequence listing applies to all sequences disclosed in a given application, whether the sequences are claimed or not. See MPEP § 2421.02.

Claim Objections

5. Regarding claims 3 & 7, the position of instructions (claim 3, line 3 / claim 7, line 2) is unclear to Examiner. Does the 'unconditional further native branch instruction' precede or follow the 'native branch instruction?' What placement is meant, using the word 'behind'?

Claim Rejections - 35 USC § 112

6. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

7. Claim 3 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. Claim 3 recites "said range of addresses" in the last line. It is unclear as to whether this refers to the previous "range of addresses" or "further range of addresses", as recited in the 3rd and 4th line. Perhaps claim 3 should be more like claim 7 and recite "a target address in a <u>first</u> range of address that does not overlap a further range of addresses....said <u>first</u> range of addresses after execution of the loop body."

Response to Arguments

8. Applicant's arguments with respect to claims 1 & 5 have been considered but are moot in view of the new grounds of rejection.

Applicant has amended claims to recite "a virtual machine interpreter, comprising a hardware pre-processor..."

Patel discloses a hardware preprocessor. See FIG. 1, #22, Hardware JAVA ACCELERATOR, #54, Program Counter separate from the processor. Col. 3, lines 35-36, "the Java hardware accelerator is shown in FIG. 1 as separate from the central processing unit..."

Claim Rejections - 35 USC § 103

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claims 1, 2, 3, 5, 6, and 7 are rejected under 35 U.S.C. 1032(a) as being unpatentable over US Patent 5,768,593 to Walters et al. in view of US Patent 6,332,215 B1 to Patel et al.

Per claim 1:

-A data processing system for executing a program of virtual machine instructions with a processor core that is arranged to execute native instructions comprising: the processor core; a memory;

(Walters: FIG. 1, col. 5, line 66-col. 6, line 10, "...computer system...includes central processing unit...and memory...")

Regarding the limitation:

-a virtual machine interpreter, comprising a hardware pre-processor including a program counter, separate from the processor core, for receiving virtual machine instructions selected dependent on program flow during execution of the program, the virtual machine interpreter being coupled to the processor core to generate native machine instructions that implement the virtual machine instructions for execution by the processor core, the virtual machine interpreter being arranged

Walters disclosed: See FIG. 2 Foreign Code, #112 (virtual machine instructions). See FIG. 4, #160 – 'Apply Hash Function to PC (program counter) for FC (foreign code / virtual machine instruction), #162, 'Is FC's PC in Hast Table?' Drawings indicate that there exists a program counter in the 'pre-processor' / cross-compiler / recompiler (col. 6, lines 25-26). The cross-

compiler/ dynamic recompiler is separate from the processor core, see FIG. 1, #102 – processor core and #116, -dynamic recompiler. Col. 3, lines 34-41, "...cross-compilation and emulation subsystem and method for converting (pre-processing), at run time, non-native code into native code immediately prior to execution of that code..." col. 4, lines 8-13; "Qualifying non-native code comprises all code that is reachable from the entry point instruction during execution of the program without going outside the compilation window and without having to first execute (A) an instruction by the interpreter...")

Walters failed to specifically disclose a hardware preprocessor. However Patel disclosed:
See FIG. 1, #22, Hardware JAVA ACCELERATOR, #54, Program Counter separate from the processor. Col. 3, lines 35-36, "the Java hardware accelerator is shown in FIG. 1 as separate from the central processing unit..." Col. 4, lines 28-33, "The Java PC can be used to obtain bytecode instructions from the instruction cache24...the Java PC is multiplexed with the normal program counter 54 of the central processing unit 26 in multiplexer 52 (receiving virtual machine instructions selected dependent on program flow during execution of the program). The normal PC 54 is not used during the operation of the Java hardware bytecode translation." As an example, col. 3, lines 35-53, "During power-on, the multiplexer 28 is set to bypass the Java hardware accelerator...In block 38, the system switches to the Java hardware accelerator mode. In the Java hardware accelerator mode, Java bytecode is transferred to the Java hardware accelerator 22, converted into native instructions then sent to the CPU for operation (receiving virtual machine instructions selected dependent on program flow during execution of the program)."

-to identify an initial virtual machine instruction from a body of successive ones of the selected virtual machine instructions, where the body is expected to be executed repeatedly;

(Walters: Col. 3, line 62- col. 4, line 3, "...cross-compiler begins compilation of a code block having an entry point instruction composed of the aforementioned next non-native instruction. If, during decoding of the entry point instruction by the cross-compiler it is determined that the entry point instruction is one of a predefined set of non-native instructions to be executed by an interpreter, then that instruction is executed by the interpreter. Otherwise, the cross-compiler continues with compilation of a block of non-native code." Walters discloses handling conditional branch instructions including loops (body executed repeatedly) at col. 5, lines 10-40.)

-to record a correspondence between the initial virtual machine instruction in the body and a memory location in the memory;

(Walters: Col. 4, lines 38-42, "...cross-compiler maintains a 'code chunk map' for indicating...code cache stores...")

-to write native instructions for the body into the memory from said memory location, the native instructions for the body being generated for virtual machine instructions starting from the initial virtual machine instruction;

(Walters: Col. 6, lines 52-58, "...hash table lookup...native code block corresponding to a non-native application code block having a specified entry point...", col. 7, lines 52-55, "If the entry

point instruction in the non-native code application does not correspond to a code block in the code cache, the recompiler begins recompilaton of the corresponding code block.", col. 13, lines 6-10, "...generating the native code for the qualifying non-native instructions by executing the code generation procedures noted in the information table for all qualifying instructions. After the native code is generated, the resulting native code block is stored in the code cache....")

-to cause the processor core to execute the native instructions for the body and
to repeat execution of the native instructions for the body by executing the written native machine instructions for the body from memory starting from said memory location.
(Walters: Col. 3, lines 45-53, "Whenever the data processor on which the cross-compiler is being used executes a non-native application, the cross-compiler is activated. The cross-compiler remains in control of program execution until execution of a native code application...is initiated...")

Walters did not disclose specifically, loop instructions, "where the body is expected to be executed repeatedly". However, Walters did provide incentive for such consideration. Col. 2, line 66-col. 3, line 5, "Since conditional branch instructions are often used at the end of execution loops in programs, conditional branch instructions are often executed large numbers of times. Also, he mentioned a type of conditional branch often found at the end of execution loops (col. 5, line 39). The inventors of the present invention have determined that (col. 3, lines 3-5) optimization of the cross-compilation of such instructions is likely to have a disproportionately beneficial affect on the execution performance of cross-compiled programs.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to have modified Walters disclosed invention to include the translation and of loop constructs (likely to be executed repeatedly) when optimizing code.

It would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Walters to include a hardware preprocessor, such as the Java accelerator, as disclosed by Patel because both references are directed towards fulfilling the need for increasing execution speed, while maintaining platform neutrality (Walters, col. 1, line 50 / Patel col. 1, line 46). While Walters pre-processes bytecodes, he uses a software technique. It is well known that software functionality may be embodied in hardware.

Per claim 2:

-the virtual machine interpreter being arranged to generate a native branch back instruction to the start of the body and placing the native branch back instruction at the end of the body in the memory.

It is well known that a block of code representing a loop will end with a branch back instruction to the start of the body. A block of code for a loop, compiled to native code will have the native branch back instruction at the end of the body.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention that the cross-compiler, as disclosed by Walters, which compiles non-native code sections into native code sections, would place a native branch back instruction at the end of the body for the purpose of control flow.

Per claim 3:

-the virtual machine interpreter being arranged to place an unconditional further native branch instruction behind the native branch instruction, the unconditional further native branch instruction having a target address in a range of addresses that does not overlap a further range of addresses in which the body is stored, the virtual machine interpreter being arranged to monitor a program counter address of the processor core and to resume selection of the virtual machine instructions and generation of native machine instructions from the selected virtual machine instructions when the program counter address enters said range of addresses after execution of the loop body.

It is well known that a block of code representing a loop will end with a branch back instruction, the instruction possibly followed by an alternate branch (If condition is true, branch to A, else branch to B…). Compiled native code, as disclosed by Walters, is stored separately from the non-native code (col.13, lines 6-10). Walter's invention will resume selection of non-native instructions (virtual machine instructions) Walter: Col. 9, line 64-col. 10, line 1, "Each exit instruction in the compiled native code stores a non-native program counter value…designates the location of the next non-native instruction…to be executed, and then returns control to the recompiler…"

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention that the cross-compiler, as disclosed by Walters, which compiles non-native code sections into native code sections, would place a native branch back instruction at the end of the body and possibly, additionally, follow the conditional branch instruction with an alternate unconditional branch instruction for the purpose of control flow..

Per claim 5:

A method of executing a program of virtual machine instructions with a processor core that is arranged to execute native instructions, the method comprising

(Walters: FIG. 1 & Col. 3, lines 34-41, "...subsystem and method for converting...", col. 6, lines 1-10, "...CPU...")

-selecting, under control of program flow, virtual machine instructions to be executed;

(Walters: Col. 3, lines 47-53, "The cross compiler remains in control...code in non-native applications...is processed by the cross-compiler...")

Regarding the limitation:

-using a hardware pre-processor, determining native instructions from the selected virtual machine instructions, to implement the selected virtual machine instructions;

Walters disclosed: Col. 3, lines 56-57, "...cross-compiler looks up the address of that next instruction in the hash table...")

Walters failed to specifically disclose a hardware preprocessor. However Patel disclosed: See FIG. 1, #22, Hardware JAVA ACCELERATOR, #54, Program Counter separate from the processor. Col. 3, lines 35-36, "the Java hardware accelerator is shown in FIG. 1 as separate from the central processing unit..." Col. 4, lines 28-33, "The Java PC can be used to obtain bytecode instructions from the instruction cache24...the Java PC is multiplexed with the normal

program counter 54 of the central processing unit 26 in multiplexer 52 (receiving virtual machine instructions selected dependent on program flow during execution of the program). The normal PC 54 is not used during the operation of the Java hardware bytecode translation." As an example, col. 3, lines 35-53, "During power-on, the multiplexer 28 is set to bypass the Java hardware accelerator...In block 38, the system switches to the Java hardware accelerator mode. In the Java hardware accelerator mode, Java bytecode is transferred to the Java hardware accelerator 22, converted into native instructions then sent to the CPU for operation (receiving virtual machine instructions selected dependent on program flow during execution of the program)."

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Walters to include a hardware preprocessor, such as the Java accelerator, as disclosed by Patel because both references are directed towards fulfilling the need for increasing execution speed, while maintaining platform neutrality (Walters, col. 1, line 50 / Patel col. 1, line 46). While Walters pre-processes bytecodes, he uses a software technique. It is well known that software functionality may be embodied in hardware.

-for a body of successive ones of the selected virtual machine instructions that is expected to be executed repeatedly;

(Walters: Col. 5, lines 10-40, Walters discloses consideration given to conditional branch instructions, including loops (bodies of successive ones of the selected virtual machine instructions that is expected to be executed repeatedly / line 40).)

-identifying an initial virtual machine instruction of the body of successive ones of the selected virtual machine instructions;
(Walters: Col. 3, lines 62-64, "...cross-compiler begins the compilation of a code block having an entry point instruction...")

-recording a correspondence between the initial virtual machine instruction and a memory location;
(Walters: Col. 4, lines 38-42, "...cross-compiler maintains a 'code chunk map' for indicating...")

-writing native instructions for the body into a memory from said memory location, the native instructions for the body being determined from virtual machine instructions starting from the initial virtual machine instruction ;
(Walters: Col. 4, lines 38-45, "...code chunk map will store...for each page that contains a non-native entry point instruction...")

-causing the processor core to execute the native instructions for the body and to repeat execution of the native instructions for the body by executing the written native

machine instructions for the body from memory starting from said memory location; and

(Walters: Col. 3, lines 37-41, "code cache for storing cross-compiled code, a hash table for locating code block in the code cache, a cross-compiler for converting blocks of non-native..." col. 3, lines 47-50, "The cross-compiler remains in control of program execution until execution of a native code application is initiated...")

Walters did not disclose specifically, loop instructions, "where the body is expected to be executed repeatedly". However, Walters did provide incentive for such consideration. Col. 2, line 66-col. 3, line 5, "Since conditional branch instructions are often used at the end of execution loops in programs, conditional branch instructions are often executed large numbers of times. The inventors of the present invention have determined that optimization of the cross-compilation of such instructions is likely to have a disproportionately beneficial affect on the execution performance of cross-compiled programs.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to have modified Walters disclosed invention to include the translation and of loop constructs (likely to be executed repeatedly) when optimizing code.

-for selected virtual machine instructions that are not expected to be executed repeatedly, providing directly to the processor core instructions to implement the selected virtual machine instructions, without storing the instructions to implement the selected virtual machine instructions in memory.

(Walters: Col. 3, lines 41-42, "...the system also includes an interpreter for emulating certain non-native instructions that are not converted into native code...", col. 3, line 64-col. 4, line 1, "If, during decoding...it is determined that...instruction is one of a predefined set of non-native instructions to be executed by an interpreter (providing directly to the processor core instructions to implement the selected virtual machine instructions)..." Walters provided for selected virtual machine instructions to be directly executed by an interpreter, not stored in memory.)

Per claim 6:

-generating a native branch back instruction to a start of the body and placing the native branch back instruction in the memory at the end of the body of successive ones of the selected virtual machine instructions that is expected to be executed repeatedly.

(See limitations addressed in claim 2 above.)

Per claim 7:

-placing an unconditional further native branch instruction behind the native branch instruction, the unconditional further native branch instruction having a target address in a first range of addresses that does not overlap a further range of addresses in which the body is stored...

(Walters disclosed branch instructions (exit instruction is inserted...execution of an instruction outside the compilation window) at col. 9, lines 55-58. Walters disclosed program counter addressing at col. 9, lines 64-67. Col. 10, lines 1-10 disclosed program counter address enters said first range of addresses after execution of the body of successive ones of repeatedly executed instructions. Interpreter or other generated instruction causes the system to switch the

mode of operation and 'enter said first range of addresses.' Walters disclosed 'unconditional further native branch instructions' and 'native branch instructions' at col. 5, lines 16-17, "conditional branch instructions...any other branch instruction...")

-the method comprising the step of monitoring a program counter address of the processor core and to resume said selecting and determining when the program counter address enters said first range of addresses after execution of the body of successive ones of the selected virtual machine instructions that is expected to be executed repeatedly.
(Walters: Col. 9, lines 64-col. 10, lines 10. Walters disclosed monitoring the program counter and switching context between non native code and native code, where the program counter enters the appropriate range of addresses after execution (of loop / successive instructions expected to be executed repeatedly).)

11. Claims 4 and 8 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 5,768,593 to Walters et al., in view of US Patent 6,332,215 B1 to Patel et al, and further in view of US Patent 5,872,978 to Hoskins.

Per claim 4:
Walters disclosed:
-the virtual machine interpreter the virtual machine interpreter recording said correspondence and writing the native instructions for the body when program flow reaches the initial virtual machine instruction.

(See limitations addressed in claim 1.)

Walters / Patel failed to address the use of 'hints' when compiling. However Hoskins suggested '-the virtual machine interpreter being arranged to receive hint information, which does not affect program flow, the hint information indicating at least said initial virtual machine instruction…' Hoskins: (col. 4, lines 56-58) "hints…act to improve the efficiency of the translation process." Col. 4, line 11, "The following are examples of suitable hints…", col. 4, lines 39-41, "BRANCH_LIKELY This tells the translator that the conditional branch instruction following this hint is more likely to be taken than not."

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to include "hints" as a technique for optimal compiling because it helps the translator to know which instructions will more likely benefit from being compiled into native code, thereby executing faster.

Per claim 8:

Walters disclosed:

-preprocessing the program to detect loop terminating with a virtual machine branch back instruction;

(Loop termination is followed by a branch back to the virtual (non-native instruction). Col. 9, line 64-col. 10, line 1, "Each exit instruction in the compiled native code stores a non-native program counter value…designates the location of the next non-native instruction…"

Walters / Patel failed to address the use of 'hints' when compiling. However Hoskins suggested

'-adding a hint to the program which identifies a target address of the virtual machine branch back instruction as the initial virtual machine instruction.' Hoskins: (col. 4, lines 56-58) "hints...act to improve the efficiency of the translation process." Col. 4, line 11, "The following are examples of suitable hints...", col. 4, lines 39-41, "BRANCH_LIKELY This tells the translator that the conditional branch instruction following this hint is more likely to be taken than not."

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to include "hints" as a technique for optimal compiling that "identifies a target address of the virtual branch back instruction as the initial virtual machine instruction" (program knows location of the non-native instruction to return to after executing native code) because ensures proper control flow.

Conclusion

12. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Note: US Patent 6,266,807 B1 to McGarity et al. (Pre-processing and directly executing of virtual language program.)

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (571) 272-3704. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen can be reached at (571) 272-3708. The fax phone number for the organization where this application or proceeding is assigned: 571-273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman Mary Steelman
03/22/2006